第42卷 第5期
2022年10月

桂林电子科技大学学报
Journal of Guilin University of Electronic Technology

Vol. 42, No. 5
Oct. 2022

基于改进蚁狮算法的虚拟机放置方法

刘耀鸿[1]，王 勇[1,2]

(1. 桂林电子科技大学 计算机与信息安全学院，广西 桂林 541004；
2. 桂林电子科技大学 广西云计算与复杂系统高校重点实验室，广西 桂林 541004)

摘 要：虚拟机放置是虚拟机整合过程中的关键步骤，虚拟机放置方法的好坏往往会影响云数据中心的资源使用效率和性能，这类问题可以通过建立多目标优化模型来进行求解。当前云数据中心存在能耗高、资源利用率较低以及资源碎片化的情况。针对上述情况，提出了一种基于MALO算法的虚拟机放置策略。通过建立多目标多约束的虚拟机放置模型，对能耗、资源利用率和资源碎片化3个方面进行优化。并且在蚁狮算法的基础上，通过改进解空间的边界变化策略和蚂蚁随机游走的位置选择策略，最后对蚂蚁位置越界进行修正，使得种群的多样性能得到更好保证，这样能更好地跳出局部最优解。基于虚拟机放置平台对MALO算法和另外4种虚拟机放置算法进行仿真实验，实验结果表明，相比于蚁狮算法、BRC算法、MBFD算法和FFD算法，MALO算法在降低能耗、提高资源利用率以及减少资源碎片化方面有一定的提升效果。

关键词：虚拟机放置；云数据中心；多目标优化；蚁狮算法；能耗

中图分类号：TP391　　**文献标志码：**A　　**文章编号：**1673-808X(2022)05-0376-08

Virtual machine placement method based on improved antlion algorithm

LIU Yaohong[1], WANG Yong[1,2]

(1. School of Computer and Information Security, Guilin University of Electronic Technology, Guilin 541004, China;
2. Guangxi Key Lab of Cloud Computing and Complex Systems,
Guilin University of Electronic Technology, Guilin 541004, China)

Abstract: Virtual machine placement is a key step in the process of virtual machine consolidation. The quality of the virtual machine placement method usually affects the resource utilization efficiency and performance of the cloud data center. Such problems can be solved by establishing a multi-objective optimization model. Currently, cloud data centers have high energy consumption, low resource utilization, and resource fragmentation. In view of the above situation, a virtual machine placement strategy based on MALO algorithm is proposed. By establishing a multi-objective and multi-constrained virtual machine placement model, the energy consumption, resource utilization, and resource fragmentation are optimized. And on the basis of the Antlion algorithm, by improving the boundary change strategy of the solution space and the location selection strategy of ants random walk, finally the position of the ants is corrected beyond the boundary, so that the diversity of the population can be better guaranteed, which can better Jump out of the local optimal solution. Based on the virtual machine placement platform, the simulation experiments of MALO algorithm and four other virtual machine placement algorithms are carried out. The experimental results show that compared to the Antlion algorithm, BRC algorithm, MBFD algorithm and FFD algorithm, the MALO algorithm has a certain improvement effect in reducing energy consumption, improving resource utilization and reducing resource fragmentation.

Key words: virtual machine placement; cloud data center; multi objective optimization; antlion algorithm; energy consumption

收稿日期：2022-04-20
基金项目：国家自然科学基金(61861013)；广西创新驱动发展专项_科技重大专项(桂科AA18118031)
通信作者：王勇(1964—)，男，教授，博士，研究方向为云计算、计算机网络技术及应用、信息安全等。E-mail：ywang@guet.edu.cn
引文格式：刘耀鸿，王勇. 基于改进蚁狮算法的虚拟机放置方法[J]. 桂林电子科技大学学报，2022，42(5)：376-383.

云计算已经成为信息行业的主要计算模式，它可以根据用户的需求来提供相关的计算资源，用户不需要关注基础设施内部的更新和维护细节[1]。这样提供资源的云服务提供商就与需要计算资源的客户建立了联系，形成了一条供应链，在虚拟化技术的基础上通过虚拟机和物理机的映射关系来满足用户的需求[2]。随着云计算的发展，虚拟化技术成为了人们关注的重点。通过虚拟化技术，可以把软件资源和硬件资源结合起来，在成本开销相对较少的情况下满足用户的计算需求[3]。

云数据中心的能源问题一直是一个至关重要的问题，它对环境和成本会产生直接的影响[4]。能源消耗、二氧化碳等问题在不断产生，使得云数据中心的能源管理面临更大的挑战。对于大型的云数据中心，能源消耗不可忽视。物理机所产生的能源消耗约占总体能源消耗的 60%。一台物理机所产生的的能耗主要由其资源上限以及部署在物理机上的虚拟机请求资源的总和所决定[5]。在通常情况下，一台空闲的物理机产生的功耗占它最大功耗的 70%[6]，因此对云数据中心的物理机功耗进行控制，减少空闲物理机的数量至关重要。Beloglazov 等[7]指出好的虚拟机放置策略能有效减少能耗，并且提高 QoS。通过优化虚拟机放置策略，可以将虚拟机更多地放置在资源利用率更高的物理机上，同时将空闲虚拟机关闭，能够最大限度地减少活动物理机的数量，从而降低能耗。除此之外，虚拟机放置的不合理会导致剩余资源不均衡，进而产生资源碎片，大量资源碎片的产生则会造成数据中心效率的低下[8]。因此，在虚拟机放置的过程中，充分考虑各台物理机的负载均衡，有助于减少资源碎片，使得数据中心的资源使用更加高效。

Jangiti 等[9]提出了一种基于资源比率的 PM 选择算法 PMNeAR，该算法会根据虚拟机请求 CPU 资源与内存资源的比例将该虚拟机放置在剩余 CPU 和内存比例最接近的物理机上，在减少资源消耗方面相比其他算法有一定的提升效果。Tarafdar 等[10]提出了一种能耗和服务质量感知的虚拟机整合算法 EQC，通过马尔可夫模型预测出空闲和过载物理机，并对它们进行迁移，旨在保证服务质量的前提下降低数据中心的能耗，该算法在降低能耗、保证服务质量和减少迁移次数方面有一定的效果。Farahnakian 等[11]提出了一种 UPBFD 算法，并使用 k-近邻回归模型对资源使用率进行预测，将过载物理机上的虚拟机进行迁移，该算法在减少能耗、迁移次数和 SLA 违反次数上有一定优势。上述近似算法在求解过程中有时间开销小的优点，但是近似算法通常是根据当前状态得到一个最优解，这个解不一定是全局最优解。文献[12-16]采用智能优化算法进行求解，智能优化算法是通过全局搜索来找到可行解。在多目标优化过程中，一般是将多个目标通过线性加权的方式整合成一个优化目标，再通过智能优化算法进行求解。Parvizi 等[12]针对能耗、资源利用率、活动物理机数量进行优化，在引入非线性凸优化解后，提出了一种非支配排序遗传算法，在与其他基础算法比较中有一定的提升。蔺凯青等[13]提出了一种基于离散蝙蝠算法的虚拟机放置算法（DBA-VMP），对经典蝙蝠算法进行改进。相比其他几种多目标优化的 VMP 算法，该算法在保证 QoS 的前提下在降低能耗和提高资源利用率两方面取得了一定的效果，但是未对 QoS 做出进一步优化。李双俐等[14]提出了一种基于 Memetic 的虚拟机放置方法，首先对虚拟机的请求进行分类，然后通过改进的 Memetic 算法进行求解，除了在降低能耗、提高资源利用率和保证 QoS 上有一定的提升效果，计算时间开销相比其他智能算法也有一定程度的减小。马小晋等[15]在模拟退火算法的基础上进行改进，提出了一种基于改进模拟退火算法的虚拟机调度方法，针对资源利用率、执行成本和负载均衡这 3 个目标进行多目标优化。为了避免陷入局部最优，引入了并行的思想，根据执行成本和执行性能生成两组解并进行迭代，最终效果有一定提升。Tang 等[16]提出了一种混合遗传算法，它不仅考虑了物理机的能耗，还考虑了通信网络的能耗，相比于普通遗传算法有着更高的性能和效率。以上这些研究大多是针对能耗、资源利用率进行优化。但在虚拟机放置的过程中，不合理的放置造成大量资源碎片的产生也同样会导致能耗较高、资源利用率低下的问题。因此，文献[17-18]主要将资源的负载均衡及资源碎片最小化作为优化目标。Li 等[17]提出了一种基于 BBO 算法的多目标虚拟机放置算法，针对 CPU、内存和带宽资源的负载均衡进行优化，同时考虑物理机之间的负载均衡及单个物理机内部的负载均衡，取得了一定的效果。Ghasemi 等[18]提出了一种基于负载均衡的多目标虚拟机放置算法，该算法使用了强化学习的方法，除了关注物理机内部的资源平衡，还关注物理机之间的负载均衡，这样能使资源分布更加均衡的同时减少运行时间。

上述研究中存在一些不足，例如在用启发式算法进行求解时，大多数只考虑了能耗、资源利用率，很少考虑资源碎片的问题，而资源碎片也是虚拟机放置中的一个关键指标，关系着虚拟机整合效果的好坏，且这类算法很容易陷入局部最优解。因此，提出一种基

于蚁狮算法的虚拟机放置算法(MALO)。通过对传统蚁狮算法进行调整,使得种群的随机性和多样性大大提升,能有效避免陷入局部最优解。

1 模型设计

1.1 虚拟机放置问题定义

虚拟机放置问题实质上可看作一个多维装箱问题,也是 NP hard 问题。假设云数据中心有 m 台虚拟机、n 台物理机,虚拟机放置问题也就是通过合理的调度策略将 m 台虚拟机放置在 n 台物理机上,且满足一定的约束条件。问题定义如下:

数据中心中 m 台虚拟机组成一个虚拟机列表 $V=\{V_1,V_2,\cdots,V_m\}$,n 台物理机组成一个物理机列表 $H=\{h_1,h_2,\cdots,h_n\}$。规定每台虚拟机请求的资源及物理机的资源包括 CPU 和内存 2 种。第 i 台虚拟机请求资源为 $R_i=(u_i,w_i)$,其中:u_i 表示第 i 台虚拟机请求的 CPU 资源;w_i 表示第 i 台虚拟机请求的内存资源。数据中心物理机的 CPU 资源容量列表为 $R_{\mathrm{cpu}}=\{c_1,c_2,\cdots,c_n\}$,其中 c_j 表示第 j 台物理机的 CPU 资源容量。数据中心物理机的内存资源容量列表为 $R_{\mathrm{mem}}=\{m_1,m_2,\cdots,m_n\}$,其中 m_j 表示第 j 台物理机的内存资源容量。并且还需要满足以下约束条件:

$$\sum_{j=1}^{n} X_{ij}=1, \tag{1}$$

$$\sum_{i=1}^{m} X_{ij}u_i \leqslant c_j, \tag{2}$$

$$\sum_{i=1}^{m} X_{ij}w_i \leqslant m_j。 \tag{3}$$

其中:$i=1,2,\cdots;m,j=1,2,\cdots,n$;$X_{ij}$ 表示虚拟机 V_i 是否放在物理机 h_j 上,它有 2 种取值 1 和 0,1 表示虚拟机 V_i 放在物理机 h_j 上,0 表示虚拟机 V_i 未放在物理机 h_j 上。式(1)表示一台虚拟机最多只能放置在一台物理机上,式(2)和式(3)表示所有虚拟机请求的 CPU 和内存资源总量不能超过当前物理机 CPU 和内存资源总量。

1.2 能耗模型

当前数据中心的高能耗问题已经引起了足够的重视,如何降低能耗已成为研究者研究的重点问题。假设当前数据中心有 m 台虚拟机和 n 台物理机,第 i 台物理机 h_i 处于满载状态时的功耗为 $h_i^{\mathrm{full_power}}$,当前的 CPU 利用率为 $h_i^{\mathrm{cpu_uti}}$。在通常情况下,可认为物理机的功耗与其 CPU 利用率呈正相关关系,能耗模型为:

$$P_i=(h_i^{\mathrm{full_power}}\alpha+h_i^{\mathrm{full_power}}(1-\alpha)h_i^{\mathrm{cpu_uti}})y_i, \tag{4}$$

$$E_{\mathrm{total}}=\sum_{i=1}^{n} p_i。 \tag{5}$$

其中:α 表示空闲物理机占满载物理机功耗的百分比,α 的范围一般在 0.6~0.7,本研究统一取 0.7;y_i 表示第 i 台物理机是否为活动物理机,它的取值为 0 或者 1,1 表示活动物理机,0 表示非活动物理机;E_{total} 表示的是数据中心的总能耗,在完成虚拟机的放置时,默认当前时刻的总功耗值为数据中心所有物理机产生的能耗。

1.3 资源使用率模型

假如当前数据中心有 m 台虚拟机和 n 台活动物理机,资源使用率包括 CPU 和内存资源的平均资源使用率 2 种,且在计算时只考虑活动物理机,而不考虑负载为 0 的物理机。资源使用率模型为

$$h_j^{\mathrm{cpu_used}}=\sum_{i=1}^{m} u_i x_{ij}, \tag{6}$$

$$h^{\mathrm{cpu_avg}}=\frac{\sum_{j=1}^{n}(h_j^{\mathrm{cpu_used}}/h_j^{\mathrm{cpu_max}})}{n}, \tag{7}$$

$$h_j^{\mathrm{mem_used}}=\sum_{i=1}^{m} w_i x_{ij}, \tag{8}$$

$$h^{\mathrm{mem_avg}}=\frac{\sum_{j=1}^{n}(h_j^{\mathrm{mem_used}}/h_j^{\mathrm{mem_max}})}{n}。 \tag{9}$$

其中:x_{ij} 表示第 i 台虚拟机是否放置在第 j 台物理机上,它有 2 个取值 0 或者 1,取值 1 表示第 i 台虚拟机放置在第 j 台物理机上,反之则取值为 0;$h_j^{\mathrm{cpu_max}}$ 和 $h_j^{\mathrm{mem_max}}$ 分别表示第 j 台物理机的 CPU 和内存资源上限;$h_j^{\mathrm{cpu_used}}$ 表示第 j 台物理机上放置的所有虚拟机请求的 CPU 资源总量;$h^{\mathrm{cpu_avg}}$ 表示所有活动物理机的平均 CPU 资源利用率;$h_j^{\mathrm{mem_used}}$ 表示第 j 台物理机上放置的所有虚拟机请求的内存资源总量;$h^{\mathrm{mem_avg}}$ 表示所有活动物理机的平均内存资源利用率。

1.4 资源平衡度模型

假设数据中心有 m 台虚拟机和 n 台物理机,为了使虚拟机的放置更加均衡,提出了一种基于比值的资源平衡度的模型,当一台物理机的 CPU 利用率和内存利用率的比值越接近于 1 时,则代表这台物理机的资源利用更加均衡。数据中心的资源平衡度模型为

$$L=\frac{\sum_{i=1}^{n}\left|\frac{h_i^{\mathrm{cpu_uti}}}{h_i^{\mathrm{mem_uti}}}-1\right|}{n}, \tag{10}$$

其中：$h_i^{cpu_uti}$ 表示第 i 台物理机的 CPU 利用率；$h_i^{mem_uti}$ 表示第 i 台物理机的内存利用率。

1.5 资源碎片模型

在当前数据中心，不合理的虚拟机放置策略可能会造成资源碎片的产生，资源碎片即物理机上 CPU 资源和内存资源不平衡，导致剩余资源无法再满足新的虚拟机请求，随着资源碎片的累积，会造成资源利用率低下及资源的浪费。资源碎片定义为

$$f_{cpu} = \sum_{i=1}^{n} h_i^{cpu_rem} y_i, \tag{11}$$

$$\text{s.t.}\ h_i^{cpu_rem} \geqslant 1 \text{ and } h_i^{mem_rem} \leqslant 512, \tag{12}$$

$$f_{mem} = \sum_{i=1}^{n} h_i^{mem_rem} y_i, \tag{13}$$

$$\text{s.t.}\ h_i^{cpu_rem} \leqslant 1 \text{ and } h_i^{mem_rem} \geqslant 512。 \tag{14}$$

其中：f_{cpu} 表示 CPU 资源碎片；f_{mem} 表示内存资源碎片；$h_i^{cpu_rem}$ 表示第 i 台物理机的 CPU 剩余量；$h_i^{mem_rem}$ 表示第 i 台物理机的内存剩余量。y_i 表示是否满足式(12)和式(14)资源碎片的条件，如果满足 y_i 为1，否则为0。

1.6 虚拟机放置目标函数

一个数据中心的虚拟机是否高效、合理，关键是能否降低能耗、提升资源利用率，并减少资源碎片的产生。低能耗和高资源利用率能使数据中心的成本降低。而虚拟机放置是否平衡同样影响着数据中心的高效性，如果服务器上资源放置不平衡，会导致资源碎片的产生。

根据上述分析，优化目标主要是最小化能耗、资源碎片，最大化资源利用率。虚拟机放置目标函数为

$$F = \min(E, L) = L + rE。 \tag{15}$$

其中：F 表示目标函数的适应度函数值，这里先将能耗和资源平衡度进行归一化，再通过加权的方式将多个目标整合成单目标；r 表示一个参数，当 r 值越大，表示能耗占的比重更大，主要考虑降低能耗，反之则表示资源平衡度占的比重更大，主要考虑虚拟机放置更加平衡，减少资源碎片。为了平衡能耗和资源平衡度的影响，将 r 的值设为0.5，此时能取得更好的放置效果。

2 基于改进蚁狮算法(MALO)的虚拟机放置方法

本质上来说，虚拟机放置问题是一个非线性的优化问题，它产生的解是离散式的，因此不适合用一般的数学方法求解。在大多数情况下，虚拟机放置问题利用启发式算法来求解，如蚁群算法、遗传算法及粒子群算法等。蚁狮优化算法(antlion optimization，简称 ALO)具有简单、调节参数少、鲁棒性比较强等优点，在求解一些复杂问题时有一定优势，但也存在不足，主要是易收敛过快，难以跳出局部最优解。因此，为了解决这类问题，需要对 ALO 算法的相关步骤进行改进。

蚁狮优化算法的步骤如下：

Step1：初始化算法的参数。种群的大小为 N，迭代次数为 T，解的维度为 dim，解的上界和下界分别为 b_u，b_l。对 N 个蚂蚁和蚁狮的位置进行随机初始化：

$$X_{i,j} = b_l + \text{rand}(b_u - b_l)。 \tag{16}$$

其中：$X_{i,j}$ 表示第 i 个种群个体的第 j 维进行初始化，$i=1,2,\cdots,N$，$j=1,2,\cdots,d$。将蚂蚁的位置和蚁狮的位置保存在矩阵中，根据目标函数计算出每个个体的适应度，并按适应度进行排序，适应度最优的位置即为精英蚁狮的位置。

Step2：通过轮盘赌策略选择蚁狮，不同位置的蚁狮被选中的概率也不一样，一般来说适应度越好，被选中的概率越大。

Step3：蚂蚁围绕轮盘赌策略选择的蚁狮和精英蚁狮进行随机游走，随机游走的方向和距离由随机函数控制。随着迭代次数的不断增加，蚂蚁随机游走的边界也在不断缩小，越来越接近最优解，边界变化如式(17)所示。最后对蚂蚁的位置进行更新，如式(19)所示。

$$I = 10^w \times \frac{t}{T}。 \tag{17}$$

$$W = \begin{cases} 2, & t > 0.1T, \\ 3, & t > 0.5T, \\ 4, & t > 0.75T, \\ 5, & t > 0.9T, \\ 6, & t > 0.95T。 \end{cases} \tag{18}$$

$$A_i^t = \frac{R_A^t + R_E^t}{2}。 \tag{19}$$

其中：I 随着迭代次数的增加而增大；t 为当前迭代次数；T 为总迭代次数；W 的值由当前迭代次数决定；A_i^t 表示第 i 个蚂蚁在第 t 次迭代中的位置；R_A^t 表示由轮盘赌选择的蚁狮在第 t 次迭代中的位置；R_E^t 表示精英蚁狮在第 t 次迭代中的位置。

Step4：在蚂蚁位置更新后会计算当前蚂蚁的适应度，若当前蚂蚁的适应度优于当前蚁狮，则蚁狮会捕获蚂蚁，并将蚁狮的位置更新为蚂蚁的位置。

Step5：迭代次数递增，若当前迭代次数达到最大

迭代次数，则输出精英蚁狮的位置，即为全局最优解；否则，重复 Step2 到 Step5，直到满足条件为止。

针对 ALO 算法容易产生的问题，提出了一种改进的蚁狮算法（modified antlion optimization，简称 MALO）。算法的主要流程如下：

输入：vmList，pmList

输出：Pos_Elite

T：最大迭代次数；iter：当前迭代次数；N：种群数量；

Pos_Ant：蚂蚁位置；Pos_Antlion：蚁狮位置；

Pos_Elite：精英蚁狮位置；

Step1：按照式(16)初始化 N 个蚂蚁和蚁狮个体的位置。

Step2：计算每个蚁狮个体的适应度函数值，并按适应度函数值从小到大排序，适应度值最小的个体位置即为 Pos_Elite。

Step3：若 iter $\leqslant T$，进行迭代，在迭代过程中边界的变化如式(20)所示，每个蚂蚁个体会围绕轮盘赌策略选择出的蚁狮以及 Pos_Elite 进行式(22)所示的随机游走。若此时 Pos_Ant$>b_u$ 或者 Pos_Ant$<b_l$，则 Pos_Ant 会按照式(23)进行更新。

Step4：计算 Pos_Ant 的适应度函数值，若小于 Pos_Antlion 的适应度函数值，则将 Pos_Antlion 更新为 Pos_Ant。

Step5：若 iter$=T$，则跳转到 Step6，否则，iter++，跳转到 Step3 继续循环。

Step6：此时 Pos_Elite 即为虚拟机放置的最终方案，输出结果。

2.1 边界自适应调整机制

在传统蚁狮算法中，蚂蚁跟随蚁狮随机游走的边界是不断变化的，随着迭代次数的增加，它的上界和下界不断收缩。但是边界变化的趋势是线性的，即所有蚂蚁个体在游走过程中边界变化趋势完全一致。这种边界变化机制不适用于云环境编码，它可能会破坏种群解的多样性，且易陷入局部最优解，不利于全局寻优。因此，提出了一种边界自适应调整机制，通过改进式(17)，使得蚂蚁进行随机游走时解的多样性更加丰富。

$$I=r+10^{w}\times\frac{t}{T}\times\left(0.5+\log_2\left(1+\frac{t}{T}\times\mathrm{rand}\right)\right), \tag{20}$$

$$r=N'_v/N'_p。 \tag{21}$$

其中：r 表示当前迭代次数待放置的虚拟机数量和物理机数量的比值；t 表示当前迭代次数；T 表示总迭代次数；w 的大小随着当前迭代次数的变化而变化，当前迭代次数越大，w 的值越大；rand 表示(0,1)区间的一个随机数。蚂蚁随机游走的边界主要由 I 来决定，并且与 I 值呈反比。I 的值是随着迭代次数的增加而动态变化的。

2.2 蚂蚁位置更新策略

蚂蚁随机游走的位置根据轮盘赌策略选择出的蚁狮及精英蚁狮的位置确定。在传统蚁狮算法中，蚂蚁最终游走的位置是轮盘赌选择的蚁狮和精英蚁狮位置的中点，这样会造成多数蚂蚁游走的趋势保持一致，破坏了种群的多样性。对蚂蚁随机游走的位置进行改进，使得在迭代的不同时期，蚂蚁游走具有不同的趋势。当迭代次数较少时，蚂蚁偏向轮盘赌选择的蚁狮进行随机游走，而在算法后期，蚂蚁更偏向精英蚁狮进行随机游走。这种策略使蚂蚁的游走更具随机性，也在一定程度上保证了种群的多样性。改进后的蚂蚁位置更新策略为

$$A_i^t=R_{\mathrm{A}}^t+(R_{\mathrm{E}}^t-R_{\mathrm{A}}^t)\times\left(\frac{t}{T}\times\mathrm{rand}+0.2\right)。 \tag{22}$$

其中：A_i^t 表示第 i 只蚂蚁在第 t 次迭代时的位置；R_{A}^t 表示第 t 次迭代时轮盘赌选择的蚁狮的位置；R_{E}^t 表示第 t 次迭代时精英蚁狮的位置；t 表示当前的迭代次数；T 表示总的迭代次数。

当蚂蚁完成随机游走后，如果此时蚂蚁的位置大于上界或者小于下界，会将蚂蚁个体移动到边界上，但这样显然不利于种群的多样性，可能会造成一些个体集中在同一位置。因此，对这种情况进行改进，如果蚂蚁个体越界，位置会按式(23)进行更新，保证随机性。

$$\mathrm{Pos_Ant}=b_l+(b_u-b_l)\times\mathrm{rand}。 \tag{23}$$

3 仿真实验及分析

用 Python 语言开发了一个虚拟机放置的仿真平台，该仿真平台可以读取数据、处理数据以及统计性能指标。提出的算法将与 FFD 算法[6]、MBFD 算法[6]、BRC 算法[19]以及 ALO 算法在能耗、资源利用率、资源碎片、活动物理机数量等几个方面进行对比。本实验的环境配置：CPU 为 2.2 GHz 六核 Intel Core i7，内存为 16 GiB，操作系统为 Mac OS，Python 版本为 Python 3.7。实验所采用的数据集是 BitBrains 数据集[20]，BitBrains 是一家真实的数据中心，并为众多公司提供云计算服务。实验场景选用的是异构的数据中心并选取了 CPU 和内存 2 种指标，数据集的物

理机配置如表 1 所示。

表 1 数据中心的物理机配置

CPU 核数	内存/GiB	物理机数量
24	32	10
24	64	20
24	128	20
48	128	10

实验中,MALO 算法的种群数量设置为 200,迭代次数为 10。待放置的虚拟机数量分别为 50,60,70,80,90 以及 100 台。数据中心的物理机数量为 60 台,且是异构的。随着待放置虚拟机的数量不断增加,使用 MALO 算法进行求解时维度也在不断增加,为了保证算法求解的精度尽可能高,实验采用分批放置,即以 10 台虚拟机为一组进行放置,以此类推。其他对比算法的参数均采用默认配置。分别对比各种算法的能耗、物理机开机数量、CPU 资源利用率、内存资源利用率、CPU 资源碎片以及内存资源碎片这几个指标。

3.1 能耗实验

实验对比了 MALO、ALO、BRC、FFD、MBFD 五种虚拟机放置算法的能耗情况以及各算法的活跃物理机数量。在对能耗进行统计时,当物理机核心数为 24、32、48、64 时,物理机对应的峰值功耗分别为 0.7、1.0、1.4、2.0 kW。

能耗实验结果如图 1 所示,MALO 算法能耗优于另外 4 种算法。当虚拟机数量为 50~90 时能耗最低,虚拟机数量为 100 时,MALO 与 ALO 能耗相同。这是因为在数据中心能耗与物理机的开机数量有很强的相关性,活跃物理机的数量越少,能耗一般也越低。但本实验是在异构的数据中心下进行的,因此削弱了这种相关性。MALO 算法在 ALO 算法上做了一些改进,有着更好的跳出局部最优解机制,在范围内能搜索到更多解,这样也就使得活跃物理机的数量减少,能耗相比其他几种算法有一定优势。

活跃物理机的数量能间接反映能耗高低,通常情况下活跃物理机数量越少,能耗越低,因此降低活跃物理机的数量是解决高能耗问题的关键。图 2 为这几种算法的活跃物理机数量,可以看出随着待放置虚拟机数量的增加,MALO 算法的活跃物理机数量是最低的,当虚拟机数量为 100 时,MALO 算法和 ALO 算法活跃物理机数量一致。这是因为 MALO 算法相比 ALO 算法有着更好的跳出局部最优解机制,有利于全局寻优。而 FFD 算法的策略是将虚拟

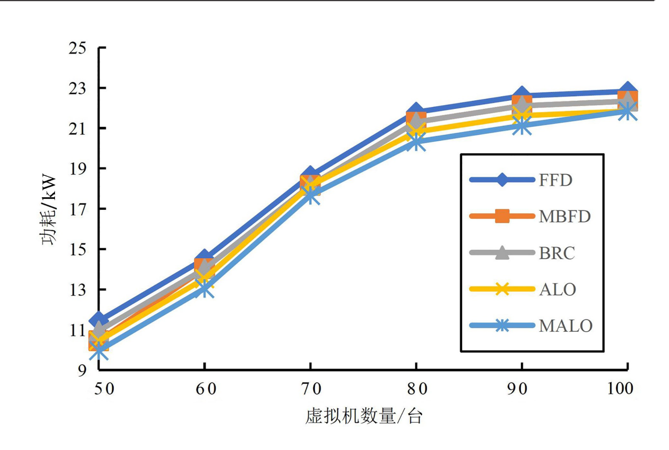


图 1 数据中心总功耗对比

机放置在遍历过程中第一台满足需求的物理机上,MBFD 算法是将虚拟机放置在能耗增加最少的物理机上,BRC 算法则是基于资源平衡因子和资源浪费得到一个合理的放置方案,这 3 种算法未尽可能地将多台虚拟机往同一台物理机上放置,从而减少物理机的开机数量。

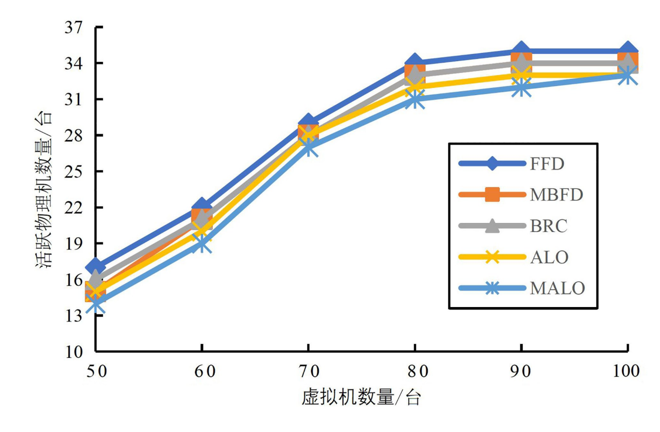


图 2 数据中心活跃物理机数量对比

3.2 资源利用率实验

资源利用率是衡量一个数据中心虚拟机放置策略好坏的关键指标,资源利用率越高意味着每台主机的资源利用更加充分,虚拟机放置策略更加合理。图 3、图 4 分别为各算法的 CPU、内存利用率的情况,在 CPU 利用率方面,在大多数虚拟机规模下 MALO 算法都具有比较明显的优势。而在内存利用率方面,当虚拟机规模在 60~90 时,MALO 算法内存利用率最高;当虚拟机数量为 50 时,内存利用率低于 MBFD 算法,当虚拟机数量为 100 时,内存利用率略低于 ALO 算法。总的来看,MALO 算法在资源利用率上能取得较好的效果,这是因为 MALO 算法相比 ALO 算法具有更好的跳出局部最优解机制,能避免陷入局部最优解,而 FFD、MBFD 及 BRC 算法没有尽可能

在一台物理机上放置更多的虚拟机，虚拟机放置较分散，活动物理机的数量较多，因此，CPU、内存利用率低于 MALO 算法。

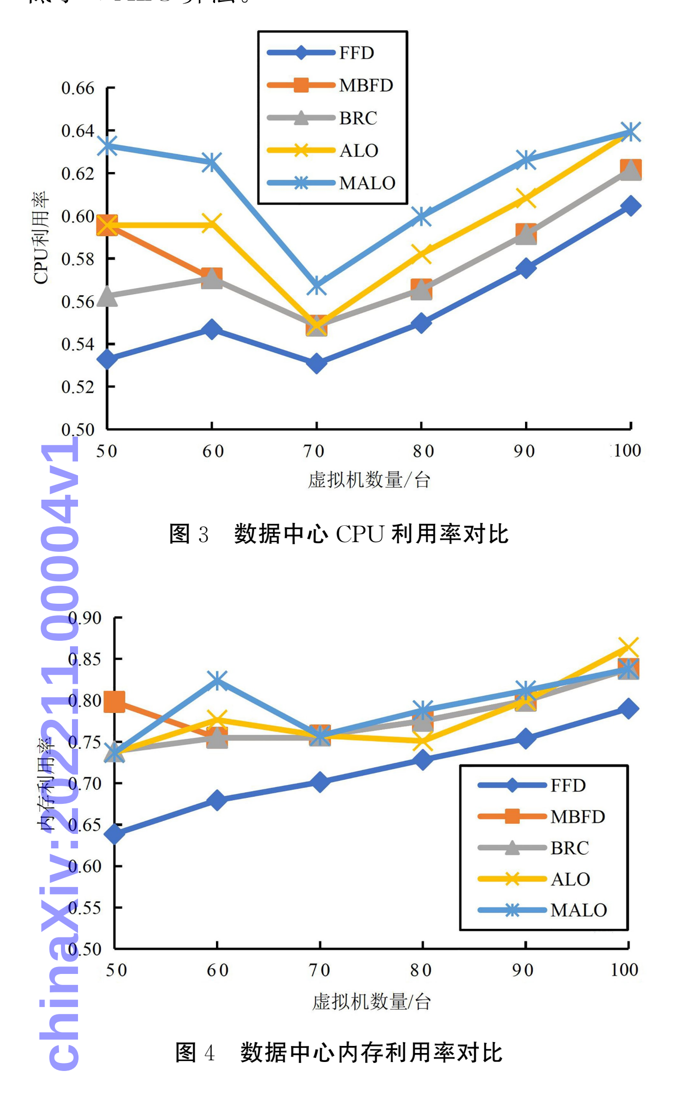


图 3 数据中心 CPU 利用率对比

图 4 数据中心内存利用率对比

3.3 资源碎片实验

资源碎片是衡量数据中心虚拟机放置是否平衡的一个重要指标。如果一台物理机上的虚拟机放置不平衡，如 CPU 资源用完而内存资源还剩很多，就会导致无法再放置新的虚拟机，资源就会被浪费掉，这也是导致物理机资源利用率低下的一个重要原因。因此，通常情况下资源碎片的数量越少，主机负载就越均衡，数据中心的资源利用越高效。通过图 5、图 6 可看出，在 CPU 碎片方面，MALO 算法与 ALO 算法效果接近，在大多数虚拟机规模下能产生更少的 CPU 资源碎片。而在内存碎片方面，MALO 算法在效果上仅次于 FFD 算法，FFD 算法在内存碎片上效果比较明显，主要是因为 FFD 的放置策略就是在遍历物理机列表的过程中找到首次满足虚拟机资源请求的物理机就进行放置，导致虚拟机放置相对比较分散，这样也就减少了内存资源碎片的产生。综合来看，MALO 算法在资源碎片方面总体上能取得一个较好的效果，这是因为 MALO 算法在进行放置时考虑了资源平衡度，使得放置后的资源平衡度尽可能小，从而减少放置不平衡情况的发生。

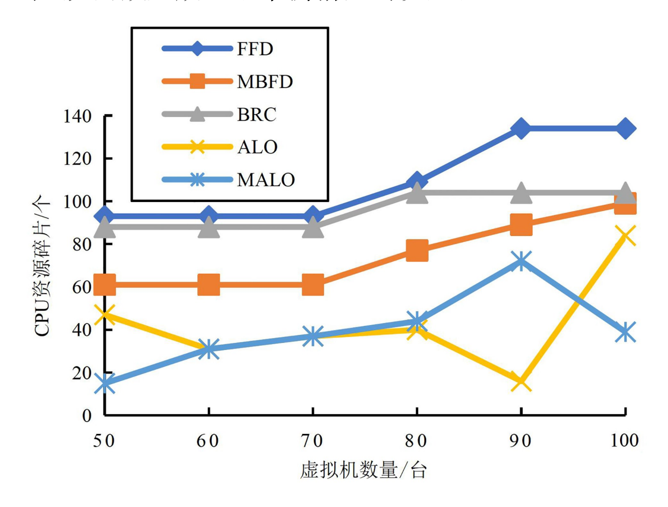


图 5 数据中心 CPU 资源碎片对比

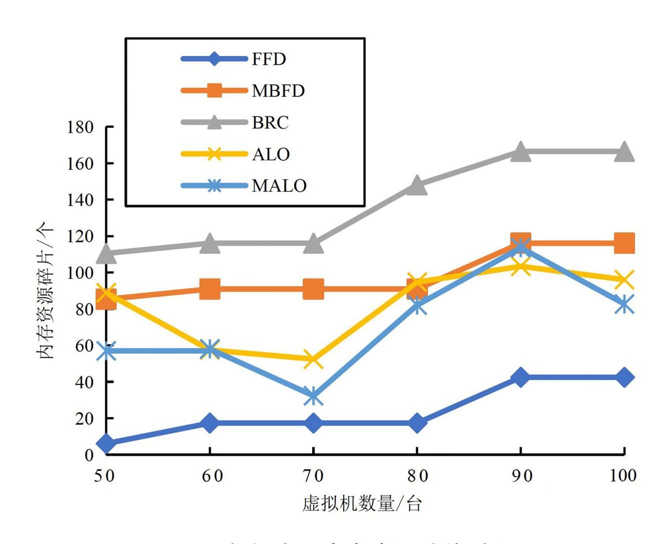


图 6 数据中心内存资源碎片对比

4 结束语

提出了一种基于改进蚁狮算法的虚拟机放置方法，针对能耗、资源利用率及资源碎片这 3 个目标进行优化。在蚁狮算法的基础上，通过在蚂蚁随机游走策略上引入自适应边界变化机制，并且改进蚂蚁随机游走时位置更新策略，使得种群的多样性大大增加，改进后的蚁狮算法有更好的跳出局部最优解机制。将此方法应用在虚拟机放置的场景下，通过与另外 4 种算法的对比实验可看出，MALO 算法在降低能耗、提高资源利用率的同时，也能在一定程度上减少资源碎片的产生。

在未来的工作中，可考虑结合蚁群算法、粒子群算法等一些智能优化算法的策略，使得算法有更好的跳出局部最优解机制。除此之外，蚁狮算法除了应用在虚拟机放置阶段，还可以应用在虚拟机迁移过程中，最后在真实环境中进一步发挥蚁狮算法的作用，提高云数据中心的效率。

参考文献：

[1] LIU Y X，ZHAO Y，DONG J，et al. I-Neat：an intelligent framework for adaptive virtual machine consolidation[J]. Tsinghua Science and Technology，2022，27(1)：13-26.

[2] MASDARI M，NABAVI S S，AHMADI V. An overview of virtual machine placement schemes in cloud computing[J]. Journal of Network and Computer Applications，2016，66：106-127.

[3] LI Y F，LI W Q，JIANG C F. A survey of virtual machine system：current technology and future trends[C]//2010 Third International Symposium on Electronic Commerce and Security. Piscataway，NJ：IEEE Press，2010：332-336.

[4] FARAHNAKIAN F，ASHRAF A，PAHIKKALA T，et al. Using ant colony system to consolidate vms for green cloud computing[J]. IEEE Transactions on Services Computing，2015，8(2)：187-198.

[5] ARROBA P，MOYA J M，AYALA J L，et al. Dynamic voltage and frequency scaling-aware dynamic consolidation of virtual machines for energy efficient cloud data centers[J]. Concurrency and Computation：Practice and Experience，2016，29(10)：494-495.

[6] BELOGLAZOV A，ABAWAJYB J，BUYYA R. Energy-aware resource allocation heuristics for efficient management of data centers for cloud computing[J]. Future Generation Computer Systems，2012，28(5)：755-768.

[7] BELOGLAZOV A，BUYYA R. Optimal online deterministic algorithms and adaptive heuristics for energy and performance efficient dynamic consolidation of virtual machines in cloud data centers[J]. Concurrency and Computation：Practice and Experience，2012，24(13)：1397-1420.

[8] ALSBATIN L，OZ G，ULUSOY A H. Efficient virtual machine placement algorithms for consolidation in cloud data centers[J]. Computer Science and Information Systems，2020，17(1)：29-50.

[9] JANGITI S，E S，JAYARAMAN R，et al. Resource ratio based virtual machine placement in heterogeneous cloud data centres[J]. Sādhanā，2019，44(12)：1-6.

[10] TARAFDAR A，DEBNATH M，KHATUA S，et al. Energy and quality of service-aware virtual machine consolidation in a cloud data center[J]. The Journal of Supercomputing，2020，76(11)：9095-9126.

[11] FARAHNAKIAN F，PAHIKKALA T，LILJEBERG P，et al. Utilization prediction aware vm consolidation approach for green cloud computing[C]//2015 IEEE 8th International Conference on Cloud Computing. Piscataway，NJ：IEEE Press，2015：381-388.

[12] PARVIZI E，REZVANI M H. Utilization-aware energy-efficient virtual machine placement in cloud networks using NSGA-III meta-heuristic approach[J]. Cluster Computing，2020，23(4)：1-23.

[13] 蔺凯青，李志华，郭曙杰，等. 云环境中多目标优化的虚拟机放置算法[J]. 计算机应用，2019，39(12)：3597-3603.

[14] 李双俐，李志华，喻新荣. 基于多目标优化的虚拟机放置方法[J]. 重庆邮电大学学报(自然科学版)，2020，32(3)：356-367.

[15] 马小晋，许华虎，卞敏捷，等. 基于改进模拟退火算法的虚拟机调度优化方法[J]. 通信学报，2018，39(增刊1)：278-287.

[16] TANG M L，PAN S C. A hybrid genetic algorithm for the energy-efficient virtual machine placement problem in data centers[J]. Neural Processing Letters，2014，41(2)：211-221.

[17] LI R，ZHENG Q H，LI X Q，et al. Multi-objective optimization for rebalancing virtual machine placement[J]. Future Generation Computer Systems，2020，105：824-842.

[18] GHASEMI A，ABOLFAZL T H. A multi-objective load balancing algorithm for virtual machine placement in cloud data centers based on machine learning[J]. Computing，2020，102(9)：2049-2072.

[19] AMMAR A M，LUO J，TANG Z，et al. Intra-balance virtual machine placement for effective reduction in energy consumption and SLA violation[J]. IEEE Access，2019，7：72387-72402.

[20] SHEN S Q，BEEK V，IOSUP A. Statistical characterization of business-critical workloads hosted in cloud datacenters[C]//IEEE/ACM International Symposium on Cluster，Cloud and Grid Computing. Piscataway，NJ：IEEE Press，2015：465-474.

编辑：梁王欢

第 42 卷 第 5 期
2022 年 10 月
桂林电子科技大学学报
Journal of Guilin University of Electronic Technology
Vol. 42，No. 5
Oct. 2022

Ceph 存储系统中节点的容错选择算法

夏亚楠，王　勇

（桂林电子科技大学 计算机与信息安全学院，广西 桂林　541004）

摘　要：Ceph 分布式系统中的数据分布算法仅将容量作为选择存储节点的标准，并未考虑存储节点的网络状态和节点负载。在副本模式下，当三副本中有存储节点并需要修复时，过高的节点负载或者网络负载会导致较大的节点修复时延。针对这个问题，给出了一种基于 Ceph 的节点容错选择（FTNSC）算法。首先利用软件定义网络技术获得实时的网络状态和节点负载信息，作为节点选择方法的数据支撑；然后建立综合考虑节点负载信息的多属性决策数学模型来确定主存储节点位置；最后通过人工蜂群算法根据与主存储节点之间的网络状态和节点性能得到最优次存储节点。实验结果表明，与现有的 CRUSH 算法相比，该算法在提高数据存储节点性能的同时，将失效数据的修复时延减少 2%～29.7%。

关键词：Ceph；软件定义网络；多属性决策；人工蜂群算法；副本放置；容错

中图分类号：TP391　　**文献标志码**：A　　**文章编号**：1673-808X(2022)05-0384-07

A fault tolerant nodes selection algorithm in Ceph storage system

XIA Yanan，WANG Yong

(School of Computer Science and Information Security，Guilin University of Electronic Technology，Guilin 541004，China)

Abstract：The data distribution algorithm in Ceph distributed system only takes the capacity as the standard for selecting storage nodes，and does not consider the network state and node load of storage nodes. In the replica mode，when a storage node in the three replicas needs to be repaired，too high node load or network load will lead to a large node repair delay. To solve this problem，a fault-tolerant node selection algorithm based on Ceph (FTNSC) is proposed. Firstly，the software defined network technology is used to obtain the real-time network state and node load information as the data support of the node selection method；Then，a multi-attribute decision-making mathematical model considering the node load information is established to determine the location of the primary storage node；Finally，the artificial bee colony algorithm is used to obtain the optimal secondary storage node according to the network state and node performance with the primary storage node. Experimental results show that compared with the existing methods，the proposed method can improve the performance of data storage nodes and reduce the data repair delay when nodes fail.

Key words：Ceph；SDN；multiple attribute decision；artificial bee colony algorithm；replica placement；fault tolerant

随着信息化的普及，各行各业每天产生的数据量都在以指数级的速度快速增长，预计到 2025 年，全球数据量相较 2016 年将增加 10 倍[1]。传统的存储模式已经无法应对海量数据的存储需求，而分布式存储系统通过廉价的商用硬件较好地解决了这一问题，目前投入商业使用的有 OpenStack Swift、Amazon EBS、Ceph 等[2]。其中，Ceph 因具有可扩展性、高性能、统一存储以及适用范围广的优势，被广泛应用于当前主流的软硬平台。

作为分布式对象存储典型代表，Ceph 最初由 Weil 等开发提出，十多年间已经有超过 100 家公司（机构）研究与使用 Ceph，其中包括欧洲原子能研究组织、Yahoo、阿里巴巴等。同时由于不同的应用场景对存储系统的关注点不同，吸引许多学者开展了一系列针对 Ceph 性能优化的研究，主要包括读写性能优化、节点工作负载优化、存储数据分布的优化等方

收稿日期：2022-04-18
基金项目：国家自然科学基金(61861013)
通信作者：王勇(1964－)，男，教授，博士，研究方向为云计算、网络信息安全、大数据、人工智能。E-mail：ywang@guet.edu.cn
引文格式：夏亚楠，王勇. Ceph 存储系统中节点的容错选择算法[J]. 桂林电子科技大学学报，2022，42(5)：384-390.